NASA/TM—2005-213844

Location Management in a Transport Layer Mobility Architecture

Wesley M. Eddy
Verizon Federal Network Systems, Cleveland, Ohio

Joseph Ishac
Glenn Research Center, Cleveland, Ohio

August 2005

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the Lead Center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at 301–621–0134

- Telephone the NASA Access Help Desk at 301–621–0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076

NASA/TM—2005-213844

Location Management in a Transport Layer Mobility Architecture

Wesley M. Eddy
Verizon Federal Network Systems, Cleveland, Ohio

Joseph Ishac
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center

August 2005

Acknowledgments

Mohammed Atiquzzaman helped to formulate the idea of a unified transport layer mobility architecture, and the IETF's Transport Area Working Group has provided feedback on the concept.

This report is a formal draft or working paper, intended to solicit comments and ideas from a technical peer group.

Available from

NASA Center for Aerospace Information
7121 Standard Drive
Hanover, MD 21076

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22100

Available electronically at http://gltrs.grc.nasa.gov

Location Management in a Transport Layer Mobility Architecture

Wesley M. Eddy
Verizon Federal Network System
Cleveland, Ohio 44135

Joseph Ishac
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract– Mobility architectures that place complexity in end nodes rather than in the network interior have many advantageous properties and are becoming popular research topics. Such architectures typically push mobility support into higher layers of the protocol stack than network layer approaches like Mobile IP. The literature is ripe with proposals to provide mobility services in the transport, session, and application layers. In this paper, we focus on a mobility architecture that makes the most significant changes to the transport layer. A common problem amongst all mobility protocols at various layers is location management, which entails translating some form of static identifier into a mobile node's dynamic location. Location management is required for mobile nodes to be able to provide globally-reachable services on-demand to other hosts. In this paper, we describe the challenges of location management in a transport layer mobility architecture, and discuss the advantages and disadvantages of various solutions proposed in the literature. Our conclusion is that, in principle, secure dynamic DNS is most desirable, although it may have current operational limitations. We note that this topic has room for further exploration, and we present this paper largely as a starting point for comparing possible solutions.

1. INTRODUCTION

In the Internet, each host interface has an IP address which represents some point on the network topology. Routing a packet destined for some IP address is accomplished by matching longer and longer prefixes of the destination address in the distributed routing table. This system works well enough for static hosts, but was not specifically designed to accommodate mobile hosts. When a mobile host moves, changing its point of attachment to the Internet, it requires a new IP address for packets to be routed to its new location.

Since IP addresses are only ephemerally useful as host identifiers, some service is required to provide more permanent host identifiers and map these into current host locations at any given time. We call this service *location management*. The literature documents a vast number of proposed mobility architectures which deal with the split between permanent identifiers and temporal locations (IP addresses) in many different ways. The common goal is to allow users and applications to use permanent identifiers to refer to hosts, and hide any complexity of the location mapping within the mobility architecture.

Mobile IP [23] hides this distinction by allowing a mobile node to be reachable at a static "home address." This is accomplished using a second "care-of address" that identifies a mobile host's current location. A "home agent" is placed within the topological area that the mobile node's home address implies. The home agent acts as an indirection point, tunneling packets arriving for the home address to the care-of address. In this mobility architecture, location management is provided by keeping the home agents updated with present care-of addresses for the mobile nodes. The permanent identifier for use by users and applications is simply the home address.

Alternatively, some mobility architectures (describing themselves as transport, session, or application layer approaches) have no concept of a home IP address, but instead use other permanent identifiers for mobile nodes. Depending on the architecture, these identifiers might be things like a name in the DNS system, a SIP address, or even a cryptographically-generated binary string. Users and applications that wish to contact a mobile host in this type of architecture must first use the appropriate means to resolve its permanent identifier into a current IP address.

The impermanence of IP addresses as host identifiers makes using permanent identifiers that are not IP addresses advantageous. For many machines on the Internet, IP addresses are dynamically obtained from a service provider using DHCP. These addresses are only temporary and often change, even when the end host is stationary. While this does not prevent them from using Mobile IP, it does prevent others from using the constantly changing IP address as a locator. Using raw IP addresses as permanent locators also only works when hosts have a "home network" for a home agent to be placed on. For many mobile devices, there may be no clear home network willing to take on this administrative burden and spare a permanent IP address from their finite pool. In addition, structural changes in organizations like businesses or universities can result in network operators readdressing network segments. While this may happen relatively infrequently in general, renumbering is not uncommon in many organizations and service providers. Dynamic host readdressing, as a privacy measure, is a proposed part of IPv6 and allows hosts to alter their address over time [20]. Thus, renumbering may become more prevalent as networks transition to IPv6.

The Mobile IP approach forces home agents into the network infrastructure. This makes networks at least slightly more difficult to configure, maintain, and troubleshoot. If every new feature had similar impositions, the utility of the network could be degraded. As conscientious architects, we try to avoid

requiring additional infrastructure when possible. If non-IP address identifiers are used, we can move the support for storage and lookup of mappings outside the network and into either end-systems or existing network services (e.g. DNS).

While not routable, identifiers that are not IP addresses may be considerably more expressive, and more directly usable for higher layer functions above routing. For instance, IP addresses are convenient for machines to work with, but not particularly natural for humans. It is much easier for people to remember a text string identifier like `www.nasa.gov` than the corresponding 32-bit IPv4 or 128-bit IPv6 address that DNS maps that name to. Systems like DNS allow us to embed *meaningfulness* in a identifier which are self-descriptive and more portable between humans (in conversation, advertisements, etc) than raw numeric IP addresses.

In a Mobile IP system, host mobility is hidden from all layers above IP. This is somewhat beneficial to higher layers, which can operate without needing any modifications to support mobility. However, some higher layer protocols and behaviors are built on the assumption of a static path. In this case, hiding mobility from them is a mistake. For instance, security policies for a host that is connected in one network might vary widely from policies that the user may expect or prefer from other networks. Common congestion control algorithms [1, 8] assume a fairly consistent path, which is fallacious assumption for many mobility scenarios. Some types of content might also be adjusted depending on a host's location, such as providing more pertinent local information (news, weather) or adjusting codecs or content items to better suit path properties like capacity, delay, loss rate, and jitter.

With similar motivations in mind, the literature contains several proposals for mobility systems that do not use IP addresses as permanent identifiers. Some of these do not provide permanent host identifiers at all, focusing only on maintaining existing connections across mobility events, while others are based entirely on the notion of separating IP addresses (locations) from host identities [21]. Notably, several systems exist for making transport layer protocols robust to changes in a host's network connection point and underlying IP address and routing changes, without requiring network layer support. For example, such systems have been proposed for TCP [26, 11, 5, 18, 3], UDP [4], SCTP [10, 15, 2], and DCCP [16]. We refer to these as transport layer mobility extensions, as they place the burden of mobility support primarily in the transport layer, with much less significant protocol changes to other layers.

Amongst these transport layer mobility systems, we can identify some common functions replicated between designs. *Movement detection* is needed to identify when a network transition has taken place and reconfigure the mobile host's addressing and routing for the new network. *Binding updates* are performed to update the address binding of existing transport connections, and a *location management* function maintains the mobile node's reachability for new connections. These are common functions that we find in most instances of what we consider to be transport layer mobility work, and we are attempting to define a common transport layer mobility architecture that can unify these approaches within a single framework [9]. This allows for redundant actions like movement detection and location management to be performed by a single module, since they are transport independent, and leave each transport layer mobility scheme to only need to provide binding updates. This makes both development and deployment of such protocols easier by narrowing the scope of problems they have to solve and minimizing the code additions and modifications required.

In this paper, we specifically discuss location management within the transport layer mobility architecture. Different mobility schemes have suggested various means for location management, and we independently evaluate and compare several of these. Section 3 discusses dynamic DNS, section 4 explores SIP, section 5 examines reliable server pooling, and section 6 covers connection-splitting approaches. Section 2 motivates why we've picked these particular methods to look at, and how we evaluate them.

2. GOALS

We have a number of goals for the location management subsystem of the transport layer mobility architecture. We describe some of the most important desires in this section, and in subsequent sections, discuss how each proposed location management solution meets or fails with regards to them.

We mostly consider only protocols that have been published in specifications by the IETF, as these are generally easier to obtain, implement, and deploy than the works of other standards bodies. We at least want to use protocols that are relatively consistent with the existing standards. The connection-splitting approaches that this document considers are not necessarily required to be "standards" as they do not change the wire-format of the protocol or its interoperability. The ideological questionability of connection-splitting practices within the traditional Internet design principles is temporarily ignored here.

2.1 Minimal Required Infrastructure or Architectural Changes

One of our most important goals is to reuse existing network infrastructure as much as possible and avoid requiring the deployment of new services or new support nodes. Additionally, we want to change the stack architecture as little as possible. That is, we would like IP to remain as small and simple a layer as possible, and we would like the interface between applications and the location manager to be as close to the current lookup system calls as possible. We especially rule out solutions that would require adding layers to the stack, as they may adversely affect interactions between the layers they displace.

It would be possible to use the existing Mobile IP technology for location management, and at the same time update transport bindings to use topologically correct addresses for their current locations, handle soft handovers, etc [14]. However, we specifically do not consider this approach within the transport layer mobility architecture. Requiring Mobile IP support in end-hosts and home agents within the network is against our principle of avoiding network infrastructure and imposing possibly ill-fitting concepts like that of a "home network" upon end users.

Specifically, we do not consider the Host Identity Protocol (HIP) [21] for location management either. Although HIP solves the fundamental problem of separating a host's location from its identity, it is unsuitable for our purposes for a couple of reasons. First, HIP requires other architectural changes like rendezvous servers, and IPsec support, which we wish to avoid imposing on the network. Also, HIP can hide mobility from

higher layers and cause the related problems explained in the previous section.

Within the transport layer mobility architecture, we attempt to create a mobility solution that places as much of the support burden outside of the network and into end hosts. If users desire mobility support, then they should be able to install software and instantly have mobility support, without requiring network managers and operators to upgrade or modify their equipment and configurations. Building a mobility solution that is mostly infrastructure-free will better empower end hosts and users than schemes like Mobile IP. We wish to scope our protocol changes so that they leave IP untouched and do not add any complexity to the processing or routing of individual packets within the network.

2.2 Secure to Remote Redirection

We call a "remote redirection" attack, one in which an attacker is able to forge an update to the location management system and cause new connections for a victim mobile node to be forwarded to some other location. The new location could be an innocent bystander, a malicious node, or non-existent host; the point is that new connections will not reach the victim itself. By remote attacks, we specifically mean ones that do not involve otherwise compromising or controlling the location management system, aside from forging some updates to it that are accepted.

The means of making the location management system robust to forged updates is itself important. For instance, cryptographic means with well accepted cryptographic primitives are preferable to less rigorous means. Public key cryptosystems are more desirable than those based on symmetric keys, as they prevent a compromise of the location manager from requiring rekeying of all managed mobile nodes. We do not see establishing key pairs as a problem, as users and hosts are likely to require these for other purposes, and preestablished keying material may be reused.

2.3 Location Privacy

A number of issues related to protection of a mobile node's identity and exact physical location are grouped together under the term "location privacy." Location privacy consists of several sub-issues, which are of widely varying importance to individuals depending on their threat models. Other documents go into greater detail on the exact problem statements and threat models [13, 12], while we provide only a brief description of how location privacy problems manifest in a transport layer mobility environment.

There are concerns over the information that a node's topologically-correct IP address can reveal about its geographic location. While a mobile node intends to update a location manager with its current information, it may be desirable to withhold information that would reveal changes in its exact location from corresponding nodes. Essentially, the corresponding node should be able to reach the mobile node without gaining any specific information about where it is. Indirection methods such as *i3* [27] or onion routing systems [7] allow a node's actual location to be shielded. However, such schemes can add a considerable amount of overhead, latency, and cost making them overkill for for the general use case.

In a transport layer mobility architecture, corresponding nodes with active connections see mobility events and location changes as a mobile node moves, without consulting the location management system. In some network layer mobility systems, like Mobile IP without route optimization, mobility events and exact location can be better hidden from corresponding nodes, since each packet travels through the location management system. The assumptions about the problems and threat models in both systems are quite different with regard to actively corresponding nodes, but similar with regards to preventing monitoring and data collection. We use the terms "lookup-based" and "indirection-based" to distinguish between whether location management is performed per connection or per packet.

Whether a lookup-based or indirection-based location manager is used, location privacy needs might require that only certain specific entities are able to determine a node's location, or that entities can be explicitly forbidden from accessing this information. Informing the mobile node of what queries have been made could also prove useful. These mechanisms can be implemented by some form of access control list and logging facility in the location manager. However, the scalability and utility of the approach is limited to only a subset of users.

Some work in network layer mobility has involved protecting the identity of a mobile node from eavesdroppers performing data collection. In this situation, the location privacy concerns are nearly identical between lookup-based and indirection-based systems. Threats here come from either passively monitoring transmissions or setting up rogue access points. In either case, we believe that good solutions to protect identity problems can occur outside of the mobility architecture, and we do not further consider these concerns in this document.

The mobile user may benefit from the ability to dynamically alter what specific network resource is being used for location management. For example, Mobile IP users depend entirely on their home network's home agents. Lookup-based location management, however, can be provided by a number of independent competitive parties. The ability for users to choose their location manager can help mitigate privacy concerns that involve the service provider's interests or motives. Also, a more distributed system makes it more difficult to perform denial of service by directing attacks at a node's location management system.

2.4 Scalable to the Future

We mean several different things under the umbrella goal of "scalable to the future." Foremost, we need a solution that works with both IPv4 and IPv6 locators, both individually and simultaneously (for dual-stack hosts). This ensures that the system will work both now and throughout the foreseeable future. We could generalize this desire to say that the system should be able to easily accommodate new types of locators, such as how DNS has been adapted to handle IPv6.

We also need a location manager that can handle the current number of mobile Internet nodes and be capable of scaling to the much larger number of mobile nodes that are expected to be present in the future. The system should be able to deal with nodes that have many locators and identifiers. This anticipates and allows for both increased end-user multihoming and more identity-based communications in the future. We would not like any artificial limitations based on present usage.

Employed security mechanisms should use cryptographic algorithms that are at least accepted by the community as being best current practices. For example, AES instead of DES. The system should also be easily retooled for new cryptographic methods, such as upgrading from RSA-based to el-

liptic curve-based primitives.

2.5 Operable Under Legacy Paradigms

Due to the value inherent in the large base of already existing application software, it is important that the initial resolution step through the location management system be performed in a way that is consistent with how current applications are written. Since nearly all present applications use a DNS resolution of a user-provided octet string into a locator set, this requirement should be easily met for any location management system that uses identifiers similar to domain names. This would at least allow the present user interfaces for inputting identifiers and most of the internal identifier handling to remain the same. In the worst case, all that we would require of applications is to adopt another mapping function than the common *gethostbyname()*, although even this level of modification is possible to avoid.

We could avoid modifying existing applications at all, by simply having the resolvers called within *gethostbyname()* support the location manager selected for transport layer mobility. For instance, on many systems other mapping services aside from DNS are already accessible through the *gethostbyname()* interface, such as "/etc/hosts" files, or the YP, NIS+, and LDAP directories. Adding support for a location manager for transport layer mobility to this system is a relatively trivial extension.

2.6 Convergence Time

A key to evaluating any mobility architecture is understanding its convergence behavior in response to updates, or how quickly old routes are globally replaced by up-to-date routes. For in-progress connections, transport layer mobility has potentially good convergence properties, since communicating end hosts receive binding updates immediately. It is also important that the location management system respond quickly to a location update in order for future connections to correctly reach a mobile node.

Since distributing directories across multiple nodes is a common way to make lookup systems robust and reduce latency, a mobile node's location information may be cached at various places in the network. There is usually a time-to-live value associated with cached entries that dictate how long they are supposed to persist. Time-to-live values for mobile nodes' locations will necessarily be quite short in comparison to other objects. These short values can make some lookup systems seem slow, because they often prevent the local caches from being consulted and result in more transactions across a higher latency path, to a busier node. For this reason, implementers have sometimes been known to be lax in obeying short time-to-live directives. This implementation practice can severely limit the *practical* usefulness of a location management system, regardless of its theoretical properties.

3. DYNAMIC DNS

The DNS protocol is a mature and ubiquitously deployed standard in the Internet. DNS provides a hierarchical naming scheme for hosts and a service for mapping those names into IP addresses [19]. Names in DNS are usually human-readable and meaningful text strings. Registration of DNS names and mappings is currently easily available to end users for nominal fees through a number of providers. In commonly used client-server applications, users make service requests for a particular host name, which is resolved through DNS into an IP address locator. Through this mechanism, end users can be completely oblivious to the underlying numeric IP addressing structure of the Internet.

The de facto standard for network programming is the BSD socket interface [6]. However, sockets themselves are bound to IP addresses and not to meaningful names. In most typical documentation for using various socket interfaces, the first step in opening a socket is resolving a DNS name into an IP address. In fact, nearly all programs that users touch perform this as the first step in any network communications, since users ask to talk to their provider's POP mail server at `pop.example.com` and their web server at `www.example.com`, rather than use the arcane 32-bit or 128-bit numbers that are the IP addresses of those hosts. With DNS, a host's location is already transparent to users and applications.

Since DNS provides a split between host identifiers and locations, which is already in common usage, it is natural to consider using DNS for location management of mobile nodes. This would allow applications that already perform DNS lookups to find a mobile node's current location, without any modifications at all to the application or the network architecture. However, the base DNS protocol does not provide any automated means for updating the system with a mobile node's current location. Updates to the directory can only be made by manually altering a file on a zone's master DNS server. This file is not typically accessible for modifications by normal end users, only by privileged network administrators.

The dynamic DNS extension [29] enables updates to the directory to be made over the network through the UPDATE addition to the DNS protocol. This enables the database to be quickly and easily changed from remote locations. A mobile host could then gain location management by sending a DNS UPDATE each time its IP address changes. To prevent DNS caches from storing stale mappings to the mobile host's old locations, it suffices to use a low time-to-live value on the resource records. The only major problem remaining with this system is that the basic DNS UPDATEs have no means for verifying that the records actually come from the proper mobile host. There needs to be some authentication mechanism that can assure DNS servers that dynamic updates only affect records that the sending parties are authorized to update.

Secure dynamic DNS can be implemented using cryptographic signatures on update messages sent to DNS servers [30]. Two flavors of secure dynamic DNS are available, based on symmetric key and public key cryptosystems. The symmetric key method is called TSIG and relies on the DNS server operator and the mobile host to have pre-exchanged the same keying bytes. Assuming that DNS service is provided by a third party, an obvious problem that arises with this approach is that it compromises the key to the DNS operator. Even if the operator themselves can be trusted, an intrusion into the operator's system can reveal all of the users' keys to an attacker, making them all vulnerable to redirection attacks.

The second method of securing dynamic DNS is referred to as SIG(0) and operates using public and private keypairs. For a mobility architecture, this has several advantages over the TSIG approach. The DNS server stores public keys and uses these to verify updates that are signed with the corre-

sponding private keys, which only the clients possess. This makes a client's key robust to any potential compromise of the DNS server. Although malicious control of the server might allow redirection attacks, it would not require rekeying after recovery. The SIG(0) system also potentially allows users to reuse preexisting public keypairs. Users only need to give the DNS operator their public key, which does not compromise their private key or the use of the keypair for other purposes. Clearly, the secure dynamic DNS extensions would be required to use DNS for location management of mobile nodes, and the SIG(0) method is preferred for this service.

For location privacy, the existing DNS system has no fine-grained control over lookups. A node's location is either openly published to the world, or not at all. Using access control lists (ACL) on inbound requests would suffice to add fine-grained capabilities to the system, but there is no suitable standard that dynamic DNS users could use for inserting and deleting ACL rules, and DNS operators would not be likely to care about providing such filtration services. In fact, they probably would not even make their clients' request logs available to them, so that clients could monitor who was tracking their location. As a location manager, DNS offers little in the line of location privacy.

The DNS directory provides the ability to store more than simple one-to-one mappings between identifiers and locators. For instance, one-to-many mappings are possible, where one identifier resolves to several addresses. This allows DNS to express hosts which are multihomed or dual-stack which could allow a node in the transport layer mobility architecture to move freely between places where distinct diverse network layers are supported. Also, the one-to-many mapping allows an identifier to refer to the addresses of multiple hosts, which could provide the same services as a mobile host in its absence. During a period of disconnection, it might be useful to have other backup hosts able to fill-in for the mobile host. DNS also allows for many-to-one mappings, where a number of identifiers resolve to the same locator. This is somewhat useful in allowing nodes to provide several contexts for humans to think of or remember them in.

Recent research into DNS response caching has shown that convergence after updates is very poor in practice [22]. This stems from implementations of caching DNS servers and DNS clients that fail to properly obey the specification and honor the lifetimes of records. As the community becomes aware of this software problem, future releases of DNS implementations should (hopefully) not exhibit this poor behavior. Until update convergence can be improved in practice, stale records floating around the network can cause problems for mobile hosts that use DNS as a location manager.

4. SESSION INITIATION PROTOCOL

The Session Initiation Protocol (SIP) [24] has been proposed for use in various mobility schemes [25, 2]. SIP is a protocol for signaling the information needed to setup connections between applications. It includes features for negotiating capabilities and locating and authenticating users. SIP is standardized by the IETF, and implementations are available from a number of vendors. SIP identifiers resemble email addresses and refer to users or services with particular grouping domains. SIP servers are currently used by a handful of applications for proxying and redirecting connection attempts. Although SIP is not yet as ubiquitous a part of the identifier to locator resolution process as DNS, it is fairly similar in basic function.

Existing literature contains good examples of how SIP can be used to support various mobility scenarios [25]. For instance, assume host A wishes to start a connection with host B. Each of these hosts has an SIP server (SIP-A and SIP-B) that stores its location state, and acts as a proxy on their behalf in performing lookups. Host A initially contacts SIP-A and tells it that it would like to start a session with host B. SIP-A then contacts SIP-B with this information. SIP-B pages host B at its current location and notifies it of A's request. The success or failure of this process propagates back through both proxies to host A. If successful at this point, hosts A and B become directly connected, and the redirection through the SIP servers ends. The lookup proxy is a reasonable feature because it removes from end hosts, the burden of performing some of the more complex lookup scenarios that SIP supports.

It has been noted that SIP can be used for paging, to provide support for micro-mobility within a domain. This is accomplished via registering the mobile host's current location as a multicast address and can also be implemented in a DNS-based system by similar means. However, hierarchically arranged SIP servers can be used to implement paging with increasing scope, which would not be possible within a purely DNS-based system. SIP can also be used for one-to-many or many-to-one mappings, as discussed in the previous section for DNS, although SIP can potentially encode more information about each locator than DNS.

SIP alone, without any transport layer modifications, can be used to provide application-layer mobility. Mobile nodes are reachable for new connections through SIP-based connection setup, but TCP-based applications will have their connections broken each time a location change takes place. Applications can handle mobility events themselves by implementing a means of restarting connections that are broken midway through an exchange. For example, the HTTP Range header can be used to specify exactly which portions of a file were not received before the mobility event and efficiently recover them. This approach requires adding significant complexity to application software and, in some cases, extending application protocols. For these reasons, it might be better to add some rebinding capability into transports rather than force connection and exchange scope re-establishment onto the transport.

Location privacy is somewhat sketchy as hosts become directly connected after the proxies set up the connection, thus revealing their location. This feature can be disabled, but requiring the SIP proxy for all communications suffers from the same scalability and vulnerability issues that a Mobile IP home agent bears. A key difference between location privacy concerns in SIP and DNS is that SIP can be configured to operate in either a lookup or indirection-based fashion.

Support for existing applications is spotty, in that a few specific applications already use SIP, but the vast majority do not. For those that do not, the mapping between SIP and DNS identifiers is not exactly direct.

Cellular SCTP is an example of a transport layer mobility scheme that uses SIP for location management [2]. Cellular SCTP leverages SCTP's built-in address management features to provide mid-connection (or association) binding updates

and relies on SIP to look up the current location of a mobile node when initiating a connection. The SIP component could easily be replaced with some other location management system.

Although it doesn't require home agents like Mobile IP, SIP proxies are needed within the network infrastructure, similar to the need for DNS servers for a dynamic DNS-based location management system. Unlike DNS, SIP servers are not yet widely and easily available to all users, although this may change. Deploying SIP services on a scale similar to DNS is an infrastructure requirement for a SIP-based location management system. The problem of converting a large number of preexisting applications to use SIP identifiers and perform SIP resolution before opening connections (and additionally specifying that all new applications will do this) is an even larger barrier. Although SIP is an attractive location manager, the required application changes seem to make it a currently impractical approach for supporting legacy applications.

SIP transactions can involve caches at multiple points. If SIP software is not well-implemented, this could lead to similar practical convergence problems that DNS currently faces. Without more operational SIP use and data-gathering, as has been performed for DNS, it is not currently possible to accurately predict if SIP will have convergence problems.

5. RELIABLE SERVER POOLING

The Reliable Server Pooling (RSERPOOL) working group within the IETF has been chartered to develop an architecture and protocols to handle management and lookup in server pools for applications with high availability and scalability requirements [28]. The basic idea is to have a way to lookup an available server from a list of those providing some service. The list is dynamically maintained as new servers become available and old ones go offline. While location management for mobile nodes is not specifically a goal, the dynamic update and lookup functions are exactly what is required of a location manager. In this case, a server pool would consist solely of a mobile node, and perhaps any alternative nodes that could suffice in the interim if the mobile node is temporarily offline.

The RSERPOOL system would insert the mobile node's new location every time it registered at a new address and remove old locations as it became unreachable at previous addresses. The RSERPOOL framework consists of two protocols: Endpoint Name Resolution Protocol (ENRP) and Aggregate Server Access Protocol (ASAP). ENRP is used within the network of servers that store the mappings, while ASAP is used by pool elements to update their entries and by clients to perform lookups. Locators are added to the pool via requests from the individual servers. Locators are removed from the pool when they are either explicitly unregistered or experience a heartbeat mechanism timeout. Mutual authentication, using TLS and certificates, in the registration and deregistration functions defends the system from remote redirection attacks. The required heartbeat responses, however, might be annoying to mobile nodes who wish to conserve power.

From a security standpoint, RSERPOOL's use of TLS is a good idea. The TLS code will generally be implemented in a library outside the RSERPOOL code, which can be easily updated or patched in response to any new vulnerabilities or to add support for new cryptographic primitives. Since many other applications use TLS, there are robust and easily obtainable implementations, which are highly likely to be maintained. The TLS protocol itself is actively maintained by the IETF and should deal promptly with both new threats and new (stronger) security primitives.

There are some fundamental differences in the designs of RSERPOOL and DNS [17]. This makes the support for the legacy resolution paradigm poor and makes RSERPOOL look like an ill fit for our goals. The identifiers provided by RSERPOOL, called "pool handles," are unformatted octet strings. Except for a few special cases, pool handles differ from domain names in several problematic ways. For instance, pool handles in RSERPOOL are not intended to be globally unique, which means resolution of them in a global location management system could be ambiguous. Also, pool handles are not necessarily long-lived identifiers, as we intend for host identities to be. Finally, RSERPOOL handles are not intended to be used, parsed, or remembered by human readers, and thus, some conversion step from human-formatted identifiers to RSERPOOL handles would be required. In many ways the goals of RSERPOOL exceed those of a simple location manager. For instance, ASAP provides means for clients to be updated when a pool changes.

In comparison to the other location management methods that this document considers, RSERPOOL is more difficult to evaluate, because its specifications have not yet been fully agreed upon and finalized. The main features that the final protocol will have are clear though, and we can evaluate RSERPOOL for our purposes based on these. The security provided to location updates is strong, but location privacy is weak. To legacy applications, RSERPOOL's problems are similar to those of SIP and perhaps even worse, given that no current applications use RSERPOOL and at least some use SIP. RSERPOOL is potentially much more fault-tolerant than any of the other location management systems that we discuss here. This makes it a good choice for scaling to a large and highly mobile Internet, but the combination of shortcomings detracts from RSERPOOL's potential for our usage.

Caching of ASAP query results has been discussed in the proposed RSERPOOL interface. Depending on implementation practices, RSERPOOL may suffer from convergence problems similar to DNS. However, it is impossible to judge at this point as development of the protocol is not yet finished, and thus it is not widely implemented or deployed.

6. CONNECTION SPLITTING

An alternative to providing an explicit location lookup service (as provided by DNS, SIP, and RSERPOOL), is to implicitly perform this at a static connection-splitting indirection node. This is the type of approach taken by MSOCKS [18] and I-TCP [3].

In these schemes, the single logical TCP connection is split into two actual TCP connections. One connection moves data between the corresponding host and the indirection point, and another connection is established between the indirection point and the mobile node. This involves making the indirection point capable of buffering data to deal with mismatches in the transfer rates between connections. The connection between the indirection point and the mobile node may run some tweaked variant of TCP that can deal with disconnections better. The required infrastructure is similar to that of Mobile IP,

although slightly more cumbersome, given that Mobile IP has no buffering requirements and may immediately and unreliably forward packets.

Signaling between the mobile node and indirection point is required when the mobile node's location changes. This could be secured either through a custom authentication step or via IPsec. In either case, it should be reasonably easy to prevent remote redirection attacks with off-the-shelf algorithms and keying methods.

Location privacy is well supported by a connection-splitting paradigm. Aside from the indirection node, no other host need be aware of the mobile hosts current location, and since the indirection node can completely split the connection, even timing analysis on connections should be unable to reveal significant clues about a mobile node's location. Since corresponding nodes always use the static indirection node as a proxy to reach the mobile host, they never have direct access to the dynamic locator mapping that DNS, SIP, or RSERPOOL provide.

In research implementations, connection splitting has been built as a transparent shim in the operating system. Applications are not aware of it, as the shim simply traps normal socket interface system or library calls and instead performs its own proxy-based versions of them. This makes the support for legacy applications good. However, the scalability to future transport protocols is somewhat poor, as both the normal kernel and shim library must be updated per transport. The scalability of a large mobile network of such hosts is also questionable, as the system's routing is inefficient and dedicated indirection nodes could be easily attacked or overloaded.

Convergence time of location updates in connection splitting is instantaneous, since a single static indirection point is used. Since there is no distributed lookup system, the cache consistency issues, which can pose potential problems for the other systems mentioned, do not exist.

7. CONCLUSIONS

Table 1 compares the location management methods that this paper describes over the listed evaluation criteria. The high-level result is that none of these technologies offers a perfect fit for our desired architecture. Dynamic DNS, secured with SIG(0) offers an attractive option, as the only facet of its evaluation that is unsatisfactory is the location privacy component. As future work, it might be possible to explore the addition of some improved form of location privacy into the DNS structure, although this may be rather difficult. Connection splitting is also somewhat attractive, although in terms of infrastructure and architectural scalability, it is less sound and less well suited to the goals of the transport layer mobility architecture.

SIP and RSERPOOL are very similar protocols by our comparison metrics. They have similar strengths and weaknesses. They are mainly questionable choices because deploying their servers would not be enough to enable them. In addition, applications would have to be rewritten to resolve locations through them rather than via DNS. Possible future work to alleviate this concern might be to formulate a protocol translation mechanism that would map DNS lookups to SIP or RSERPOOL lookups. This might be possible to do in a relatively safe and straightforward way, and would allow the location book-keeping to be done in a different way than dynamic DNS, while maintaining compatibility with existing applications.

8. REFERENCES

[1] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control, April 1999. RFC 2581.

[2] I. Aydin and C. Shen. Cellular SCTP: A Transport-Layer Approach to Internet Mobility. In *12th International Conference on Computer Communications and Networking (ICCCN 2003)*, October 2003.

[3] A. Bakre and B. R. Badrinath. I-TCP: Indirect TCP for Mobile Hosts. *15th International Conference on Distributed Computing Systems (ICDS'95)*, 1995.

[4] K. Brown and Suresh Singh. M-UDP: UDP for Mobile Cellular Networks. *Computer Communication Review 26 (4)*, October 1996.

[5] K. Brown and Suresh Singh. M-TCP: TCP for Mobile Cellular Networks. *Computer Communication Review 27 (5)*, October 1997.

[6] D. Comer and D. Stevens. Internetworking with TCP/IP, Volume III, Client-Server Programming and Applications (Windows Sockets Version). Prentice-Hall, Inc, 1997

[7] D. Goldschlag, M. Reed, and P. Syverson. Onion Routing for Anonymous and Private Internet Connections. *Communications of the ACM 42 (2)*, February 1999.

[8] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification, January 2003. RFC 3448.

[9] W. Eddy, J. Ishac, and M.Atiquzzaman. An Architecture for Transport Layer Mobility. Internet-Draft (work in progress), August 2004.

[10] S. Fu, M. Atiquzzaman, L. Ma, W. Ivancic, Y. Lee, J. Jones, and S. Lu. TraSH: A Transport Layer Seamless Handover for Mobile Networks, January 2004. University of Oklahoma Technical Report OU-TNRL-04-10.

[11] D. Funato, K. Yasuda, and H. Tokuda. TCP-R: TCP Mobility Support for Continuous Operation. In *IEEE International Conference on Network Protocols*, October 1997.

[12] W. Haddad, E. Nordmark, F. Dupont, M. Bagnulo, S. Park, B. Patil, and H. Tschofenig. Privacy for Mobile and Multi-homed Nodes (MoMiPriv) Formalizing the Threat Model. Internet-Draft (work in progress), February 2005.

[13] W. Haddad, E. Nordmark, F. Dupont, M. Bagnulo, S. Park, and B. Patil. Privacy for Mobile and Multi-homed Nodes: MoMiPriv Problem Statment. Internet-Draft (work in progress), February 2005.

[14] S. J. Koh, H. Y. Jung, S. H. Kim, and J. S. Lee. SCTP with Mobile IP for IP Mobility Support. Internet-Draft (work in progress), February 2003.

[15] S. J. Koh, H. Y. Jung, and J. H. Min. Mobile SCTP for IP Mobility Support in Transport Layer. *Proceedings of CIC (Cellular and Intelligent Communications)*, October 2003.

[16] E. Kohler. Datagram Congestion Control Protocol Mobility and Multihoming. Internet-Draft (work in progress), July 2004.

[17] J. Loughney, M. Stillman, Q. Xie, R. Stewart, A. Silverton. Comparison of Protocols for Reliable Server Pooling. Internet-Draft (work in progress), July 2004.

[18] D. Maltz and P. Bhagwat. MSOCKS: An Architecture for Transport Layer Mobility. In *IEEE INFOCOM*, 1998.

	Infrastructure	Update Security	Location Privacy	Future Scalability	Legacy Support	Convergence Time
DNS	preexisting	feasible with SIG(0)	open	possible	yes	currently poor
SIP	partial	feasible	configurable	possible	sparse	unknown
RSERPOOL	deployment required	yes	open	questionable	poor	unknown
Connection-Splitting	deployment required	feasible	yes	questionable	yes	good

1: Comparison of Location Management Technologies

[19] P. Mockapetris. Domain Names - Implementation and Specification, November 1987. RFC 1035.

[20] T. Narten, R. Draves. Privacy Extensions for Stateless Address Autoconfiguration in IPv6, January 2001. RFC 3041.

[21] P. Nikander, J. Ylitalo, and J. Wall. Integrating Security, Mobility, and Multi-homing in a HIP Way. In *Proceedings of Network and Distributed Systems Security Symposium (NDSS'03)*, February 2003.

[22] J. Pang, A .Akella, A. Shaikh, B. Krishnamurthy, S. Seshan. On the Responsiveness of DNS-based Network Control. Proceedings of the Internet Measurement Conference 2004, October 2004.

[23] C. Perkins. IP Mobility Support for IPv4, January 2002. RFC 3220.

[24] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol, June 2002. RFC 3261.

[25] H. Schulzrinne and E. Wedlund Application-layer Mobility Using SIP In *ACM SIGMOBILE Mobile Computing and Communications Review 4(3)*, July 2000.

[26] A. Snoeren and H. Balakrishnan. An End-to-End Approach to Host Mobility. In *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, August 2000.

[27] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, S. Surana. Internet Indirection Infrastructure. *Proceedings of ACM SIGCOMM*, August 2002.

[28] M. Tuexen, Q. Xie, R. Stewart, M. Shore, L. Ong, J. Loughney, and M. Stillman. Requirements for Reliable Server Pooling, January 2002. RFC 3237.

[29] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound. Dynamic Updates in the Domain Name System (DNS UPDATE), April 1997. RFC 2136.

[30] B. Wellington. Secure Domain Name System (DNS) Dynamic Update, November 2000. RFC 3007.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 2005	Technical Memorandum

4. TITLE AND SUBTITLE

Location Management in a Transport Layer Mobility Architecture

5. FUNDING NUMBERS

WBS–22–184–10–06

6. AUTHOR(S)

Wesley M. Eddy and Joseph Ishac

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–15218

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM—2005–213844

11. SUPPLEMENTARY NOTES

Wesley M. Eddy, e-mail: Wesley.M.Eddy@nasa.gov, Verizon Federal Network Systems, 21000 Brookpark Road, Cleveland, Ohio 44135; and Joseph Ishac, e-mail: Joseph.A.Ishac@nasa.gov, NASA Glenn Research Center. Responsible person, Joseph Ishac, organization code RCN, 216–433–6587.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category: 62

Available electronically at http://gltrs.grc.nasa.gov
This publication is available from the NASA Center for AeroSpace Information, 301–621–0390.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Mobility architectures that place complexity in end nodes rather than in the network interior have many advantageous properties and are becoming popular research topics. Such architectures typically push mobility support into higher layers of the protocol stack than network layer approaches like Mobile IP. The literature is ripe with proposals to provide mobility services in the transport, session, and application layers. In this paper, we focus on a mobility architecture that makes the most significant changes to the transport layer. A common problem amongst all mobility protocols at various layers is location management, which entails translating some form of static identifier into a mobile node's dynamic location. Location management is required for mobile nodes to be able to provide globally-reachable services on-demand to other hosts. In this paper, we describe the challenges of location management in a transport layer mobility architecture, and discuss the advantages and disadvantages of various solutions proposed in the literature. Our conclusion is that, in principle, secure dynamic DNS is most desirable, although it may have current operational limitations. We note that this topic has room for futher exploration, and we present this paper largely as a starting point for comparing possible solutions.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Computer networks; Protocol (computers); Mobility	14
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	